

## VisComposer

目前设想将论文/实现分为三个部分：

论文	实现
<p>Scenegraph 的具体定义 类似于 VEGA-Lite 中这样的形式：</p> <div><math display="block">selection := (name, type, predicate, domain range, event, init, transforms, resolve)</math></div> <p>具体的定义出 Scenegraph：</p> <ul style="list-style-type: none"><li>● <b>基础单元</b>：基于代理的视觉映射绑定</li><li>● <b>组成结构</b>：按行、按列、组合、聚合等；用于实现静态的图表</li><li>● <b>控制</b>（状态机）：按数据类型或其他设定的条件分配视觉映射方式等；用于添加动画、交互</li></ul>	<p>一套底层的渲染框架 可以将特定格式（如 JSON）的定义绘制成最终结果的渲染器 需要定义完善的模板格式</p>
<p>基于步骤的交互创造基于状态的结果 基于步骤的创建过程更为直观，但容易出现 BUG，如不可逆等 举例来说： 用户希望点击选中目标，点击空白处取消，于是需要完成两个函数：</p> <pre>obj.onClick():     this.color='red'     selected=this canvas.onClick():     selected.color=XXX</pre> <p>但当有其他非点击情况会影响到选择时（如数据刷新了），这些函数并不会被自动调用，而且上述的 <code>selected</code> 变量也可能未被正确重置，导致颜色出错 正确的写法应当是更新是根据状态决定： <code>obj.update():if(obj.selected)...</code> 但基于状态的写法相对不直观，需要提前做好铺垫（设置状态和转移） 目标是能够通过基于步骤的交互产生基于状态的模板（通过上文定义“<b>控制</b>”机制）；类似于计算理论中的 NFA 转 DFA</p>	<p>定义一套方便直观交互，主要包含两个部分功能：</p> <ol style="list-style-type: none"><li><b>1. 数据绑定</b> 数据基于树状结构进行分发，但部分节点可能需要更为复杂的绑定，例如一个处于第三层的聚合节点可能是如下数据绑定：<math display="block">Obj(i_1, i_2, i_3) = \sum_{j=1}^{i_2} D(i_1, j, i_2)</math><p>其基本结构依然基于树状，但有其他结构存在。需要提供直观的方式让用户能选择到这样一些抽象结构。</p></li><li><b>2. 状态生成</b> 如上所述，通过简单的交互指定（如：我需要在添加一个点击相应）生成对应的<b>控制</b>结构（如：分别生成点击前后的两个状态）。需要一定的推荐（如 VisComposer 视频中对对角线判定的推荐）和自动绑定（如只需要改颜色，则其他对应映射自动复制）功能。</li></ol>

通过模板化、快速的可视化构建和数据绑定，能够将可视分析中的交互反馈应用到数据变换和视觉映射更大的范围中。不再局限于修改一些既定参数，而是可以随意更改数据变换方式、视觉绑定和视图的选择等等。使得可视分析的循环更加完整。 如果可以，再添加一些应用实际数据进行可视分析的样例体现系统的实际作用	实现模板化。需要不光能在模板不变的情况下替换数据，同时也可以试图在数据不变的情况下替换模板。
--	--

总体来说，我和万琪讨论过，认为之前 VisComposer 视频中的一些交互手段非常不错，将其完善成完整的构架体系就足够好了。

## 博士生中期答辩

题目：数据可视化分析架构

具体内容见 PPT

## 下周任务

- 周一 博士生中期答辩
- 项目任务安排

我开始实现底层渲染器

胡万琪、刘良军：先完成电商项目，然后开始搭建网页整体框架和界面

魏雅婷：先完成手头项目，然后可以跟万琪做一下网站的搭建；之后再选择做电网数据或者 VisComposer 都行

- 周五 出发去 VisWeek。目前考虑所有人坐机场大巴一同前往浦东机场